

Документация по поддержке жизненного цикла ПО «MsVision»

1. Введение

Настоящая документация описывает этапы проектирования, разработки, тестирования, внедрения и сопровождения программного обеспечения MS Vision. В ней приведены основные методики и процедуры, необходимые для стабильного функционирования системы, обеспечения качества и оперативного реагирования на запросы пользователей. Документ ориентирован на команду разработчиков, тестировщиков, системных и DevOps-инженеров, а также сотрудников технической поддержки.

Основная модель разработки схематически основана на следующем процессе, который можно охарактеризовать как водопад.

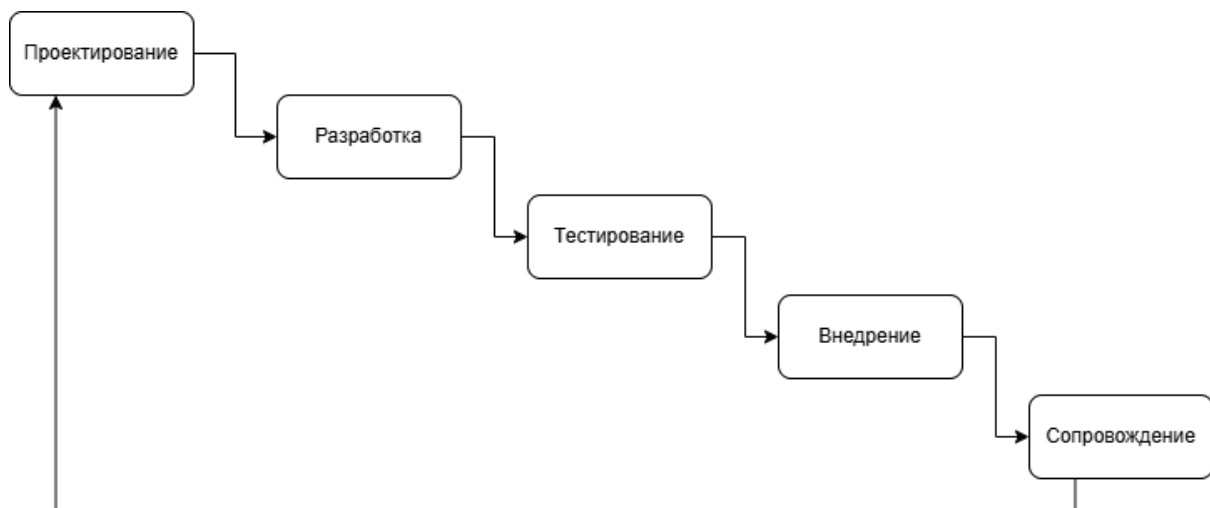


Рис. 1 Жизненный цикл программного продукта

2. Проектирование

В этой фазе определяются архитектурные и технологические решения, создаются первичные макеты интерфейса и формируется первичная проектная документация.

2.1. Архитектурное проектирование системы

- Определение модульной структуры (Backend, Frontend, Android-клиент, сервисы интеграции).
- Выбор основных технологий и инструментов: Laravel 11.x (PHP) для серверной части, JavaScript для веб-интерфейса, Kotlin (Android SDK API 27+) для мобильного клиента, PostgreSQL 12+ для СУБД, Redis 6+ для кэша и очередей.

2.2. Создание прототипов и макетов пользовательского интерфейса

- Разработка wireframes основных экранов (обзор плейлистов, мониторинг панелей, загрузка контента).
- Подготовка mockup'ов в графическом редакторе (Figma) с указанием расположения элементов управления и полей ввода.

2.3. Разработка технической документации

- Описание функциональных модулей (управление контентом, формирование расписаний, мониторинг).
- Составление спецификаций API для взаимодействия веб-интерфейса и мобильного клиента с сервером.
- Подготовка требований к безопасности (аутентификация, авторизация, хранение данных).

3. Разработка

На этапе разработки создаётся исходный код в соответствии с утверждёнными архитектурными решениями и проверяется качество кода.

3.1. Написание кода по проектной документации

- Backend: реализация REST-API на Laravel (модули контента, плейлистов, мониторинга).
- Frontend: реализация интерфейса на Vue.js 3 с авторизацией, загрузкой и управлением данными.
- Android-клиент: разработка APK-приложения на Kotlin с функционалом отображения контента и мониторинга статуса.

3.2. Регулярные код-ревью и модульное тестирование

- Проведение peer-review каждого pull request в Git-репозитории.
- Написание и выполнение unit-тестов на PHPUnit и PestPHP (контроллеры, модели, сервисы).
- Использование статического анализа кода (PHPStan, ESLint) для поддержания стандартов качества.

3.3. Интеграция компонентов

- Сборка Docker-контейнеров: PHP-FPM (Backend), Nginx (веб-сервер), PostgreSQL (СУБД), Redis (кэш/очереди)
- Настройка Docker Compose для локального, тестового и рабочего развёртывания.
- Тестирование взаимодействия сервисов (запросы API, отправка расписаний, передача скриншотов и других функций).

4. Тестирование

Цель этапа — подтвердить соответствие системы функциональным, нагрузочным и требованиям безопасности.

4.1. Функциональное тестирование

- Проверка выполнения основных сценариев: загрузка медиа, создание и публикация плейлистов, регистрация панелей, мониторинг статуса.
- Тестирование CRUD-операций для контента, плейлистов и пользователей.

4.2. Регрессионное тестирование

- Автоматизированное прогон тестовых наборов после каждого релиза.
- Проверка отсутствия новых дефектов при добавлении изменений.

4.3. Нагрузочное тестирование

- Имитация большого количества одновременных подключений к API и веб-интерфейсу
- Оценка устойчивости Redis и СУБД при интенсивной работе (массовая отправка логов с устройств).

4.4. Подготовка тестовой документации и отчетов

- Формирование отчётов о покрытиях функциональных сценариев, результатах нагрузочных замеров.
- Описание найденных дефектов с приоритетом (критические, средние, незначительные).

5. Внедрение

Подготовка и запуск системы в рабочей инфраструктуре.

5.1. Подготовка к развертыванию

- Сборка финальных Docker-образов и загрузка их в корпоративный реестр.
- Настройка серверной инфраструктуры (виртуальные машины): установка Docker Engine, настройка сетевой безопасности, резервного копирования.

5.2. Настройка окружения

- Формирование production- и staging-окружений с одинаковыми конфигурациями (переменные `.env`, параметры Docker Compose).
- Настройка CI/CD-пайплайнов (GitLab CI) для автоматической сборки и деплоя.

5.3. Обучение пользователей и администраторов

- Проведение видеоконференций и подготовка пользовательских инструкций (PDF-документы) по работе с интерфейсом: загрузка контента, создание плейлистов, мониторинг панелей.
- Обучение ИТ-персонала работе с Docker-контейнерами, базой данных и резервным копированием.

5.4. Запуск системы и переход на эксплуатацию

- Выпуск релиза в production: переключение DNS-записей (маршрутизация запросов), проверка доступности сервисов.
- Мониторинг состояния в первые недели после старта (обработка инцидентов, корректировка конфигураций).

6. Сопровождение

Обеспечение стабильной работы системы после внедрения и непрерывное улучшение продукта.

6.1. Мониторинг и поддержка производительности

- Настройка инструментов мониторинга (Zabbix) для отслеживания метрик: загрузка CPU, оперативная память, свободное место на диске.
- Регулярный анализ логов для своевременного выявления проблем.

6.2. Поддержка пользователей

- Организация службы техподдержки: выделенный канал (email, телефон) для приёма обращений.
- Обработка заявок в системе тикетов (DevProm): классификация, эскалация и контроль SLA.

6.3. Сбор обратной связи и планирование улучшений

- Проведение опросов пользователей и анализ статистики использования модулей.

- Формирование бэклога предложений по улучшению функциональности (доработки UI, новые форматы контента, интеграции).

6.4. Регулярные обновления и выпуск исправлений

- Выпуск патчей безопасности и исправление критических дефектов (обновление версий Laravel, PHP, зависимостей).
- Разработка новых релизов с учётом отзывов: добавление функций, оптимизация производительности, поддержка новых платформ и форматов.

6.5. Поддержка совместимости

- Адаптация системы к изменениям внешних сервисов (API S3), обновление зависимостей и библиотек для обеспечения совместимости.
- Тестирование при переходе на новые версии Android или изменения сетевой инфраструктуры.

7. Данные о персонале, задействованном в жизненном цикле разработки продукта

Ниже приведены сведения о персонале, задействованном в процессе сопровождения продукта:

Направление	Технологический стек	Численность (чел)
Разработчик серверная часть	PHP, Laravel, Linux, Docker	3
Разработчик Android	Kotlin, Java, Hilt/Dagger, Room, Media3 ExoPlayer, Retrofit2, Firebase	2
Системный Аналитик	UML, BPMN, ER-диаграммы, Confluence	2
Тестировщик	PHPUnit, PestPHP	1

Специалист службы технической поддержки	Знание MS Vision, работа с DevProm, базовые навыки администрирования	2
---	--	---

Контактная информация:

- **Наименование организации:** ООО «Медиа Страйк».
- **Юр. адрес:** 603105, г. Нижний Новгород, ул. Ошарская, д. 95, помещение ПЗ, 4 этаж
- **ОГРН:** 1105260001890
- **ИНН:** 5260271675
- **КПП:** 526201001

Контактная информация службы технической поддержки

- **Сайт:** <https://msvision.ru>
- **Телефон:** +7 831 233 21 89
- **Email:** info@msvision.ru